

# **FERRAMENTA WEB DE SUPORTE A AVALIAÇÃO DE SOFTWARE COM A METODOLOGIA CERTICS**

Acadêmico: Vinícius Ferneda de Lima

[vinicius.ferneda@gmail.com](mailto:vinicius.ferneda@gmail.com)

Orientador: Prof. Everaldo Artur Grahl

[everaldo.grahl@gmail.com](mailto:everaldo.grahl@gmail.com)

# Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Especificação
- Implementação
- Operacionalidade da implementação
- Resultados e discussões
- Conclusões e sugestões

# Introdução

- Esta ferramenta visa apoiar a avaliação de um consultor sobre um software da organização solicitante.
- Permite aos professores da área de qualidade de software uma opção para aulas práticas.
- São restritas as soluções para adoção da metodologia CERTICS.

# Objetivos

- O objetivo geral deste trabalho é desenvolver uma ferramenta web que possa apoiar a avaliação de software usando a metodologia CERTICS.
- Os outros objetivos são:
  - disponibilizar um ambiente que permita configurar a metodologia de avaliação da CERTICS;
  - agilizar o cálculo dos resultados de uma avaliação e possibilitar a emissão de relatórios detalhados com a consequente resultância das avaliações do software;
  - permitir que os avaliadores consigam avaliar as evidências registradas pela empresa que está sendo analisada.

# Fundamentação Teórica

- O que é CERTICS?
  - é uma certificação voluntária que identifica, credencia e diferencia software resultante de desenvolvimento e inovação tecnológica realizados no País.
- Foi criada em 2013 pelo Ministério da Ciência, Tecnologia e Inovação (MCTI) como parte do programa TI Maior.

# Fundamentação teórica

- A certificação serve de instrumento às organizações que buscam qualificação em compras públicas a partir da introdução de margem de preferência a detentores de tecnologia desenvolvida no País.
- Certifica o software de Organizações desenvolvedoras instaladas no território nacional.

# Fundamentação teórica

- Quem se beneficia?
  - Empresas desenvolvedoras de Software.
  - Compradores responsáveis por licitações.
  - Entidades compradoras.
- A metodologia CERTICS é considerada:
  - Inclusiva.
  - Ágil.
  - Acessível.

# Fundamentação teórica

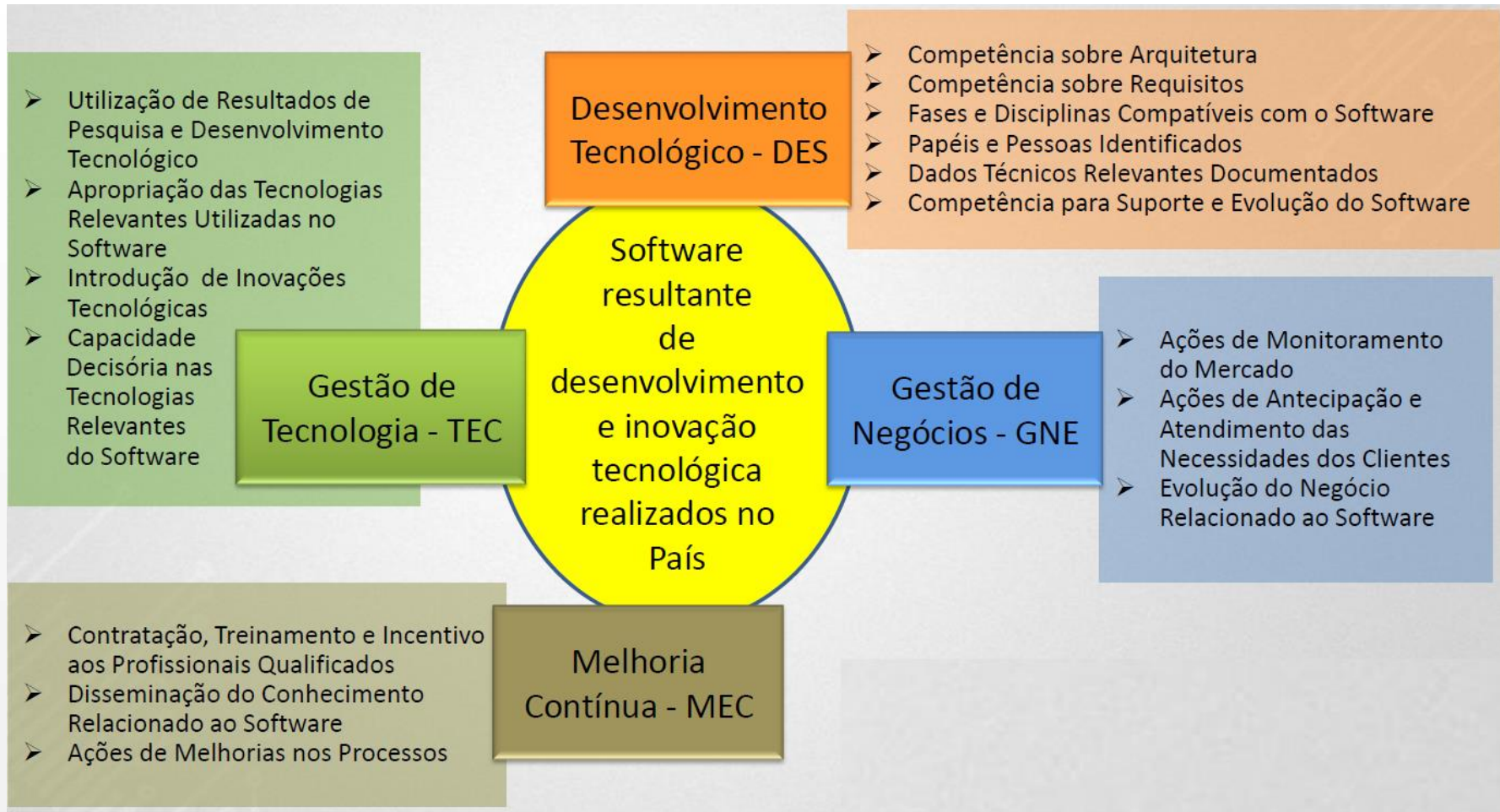
- A metodologia segue as seguintes diretrizes:
  - A avaliação é do software, não da empresa, e é baseada na análise dos processos utilizados no software.
  - A metodologia é baseada na Norma ABNT NBR ISO/IEC 15504 para avaliação de processo.
  - Um novo conceito demanda um novo modelo de referência e um novo método para avaliação.



# Fundamentação teórica

- A avaliação é baseada em evidências que demonstram o grau de atendimento de resultados esperados em quatro áreas de competências.
- As quatro áreas de competências são as seguintes:
  - Desenvolvimento Tecnológico (DES).
  - Gestão de Tecnologia (TEC).
  - Gestão de Negócios (GNE).
  - Melhoria Continua (MEC).

# Fundamentação teórica



# Fundamentação Teórica

- O resultado de uma avaliação na metodologia CERTICS é definido a partir da pontuação de cada resultado esperado. A pontuação é classificada da seguinte forma:
  - completamente atendido (*fully* – F);
  - largamente atendido (*largely* – L);
  - parcialmente atendido (*partially* – P);
  - não atendido (*not* – N).

# Trabalhos Correlatos

- WISE: Uma Abordagem de Ferramenta de Software Para o Auxílio no Modelo de Avaliação do MPS.BR.
- Ambiente Web de Suporte ao Processo de Avaliação da Qualidade de Produtos de Software.
- CERTICSys.

# Requisitos

Baseado nos trabalhos pesquisados correlatos e na descrição da metodologia CERTICS foram identificados os seguintes requisitos:

- permitir o cadastro de usuário (Requisito Funcional – RF);
- permitir a definição de áreas de competência (RF);
- permitir a definição dos resultados esperados (RF);
- permitir o cadastro de avaliador (RF);

# Requisitos

- permitir o cadastro de organização (RF);
- permitir o cadastro de profissionais da organização (RF);
- permitir o cadastro de softwares (RF);
- permitir o cadastro de evidências (RF);
- permitir o cadastro de anexos (RF);
- permitir a definição de escalas de pontuação (RF);
- permitir a criação de uma avaliação de software (RF);

# Requisitos

- permitir o vínculo de evidências em uma avaliação (RF);
- permitir o registro de observações sobre uma avaliação (RF);
- permitir o cálculo do resultado de uma avaliação (RF);
- permitir a geração de um gráfico de atendimento das áreas de competência (RF);
- permitir a geração de um relatório do resultado final da avaliação detalhado (RF);

# Requisitos

- utilizar a linguagem de programação Java 1.7 (Requisito Não-Funcional - RNF);
- utilizar o ambiente de desenvolvimento Eclipse 4.3 (RNF);
- utilizar JSF 2.0 com o *framework* PrimeFaces para que a ferramenta ofereça suporte web (RNF);
- utilizar JasperReports (RNF);
- utilizar iReport (RNF);
- utilizar Banco de dados PostgreSQL (RNF).



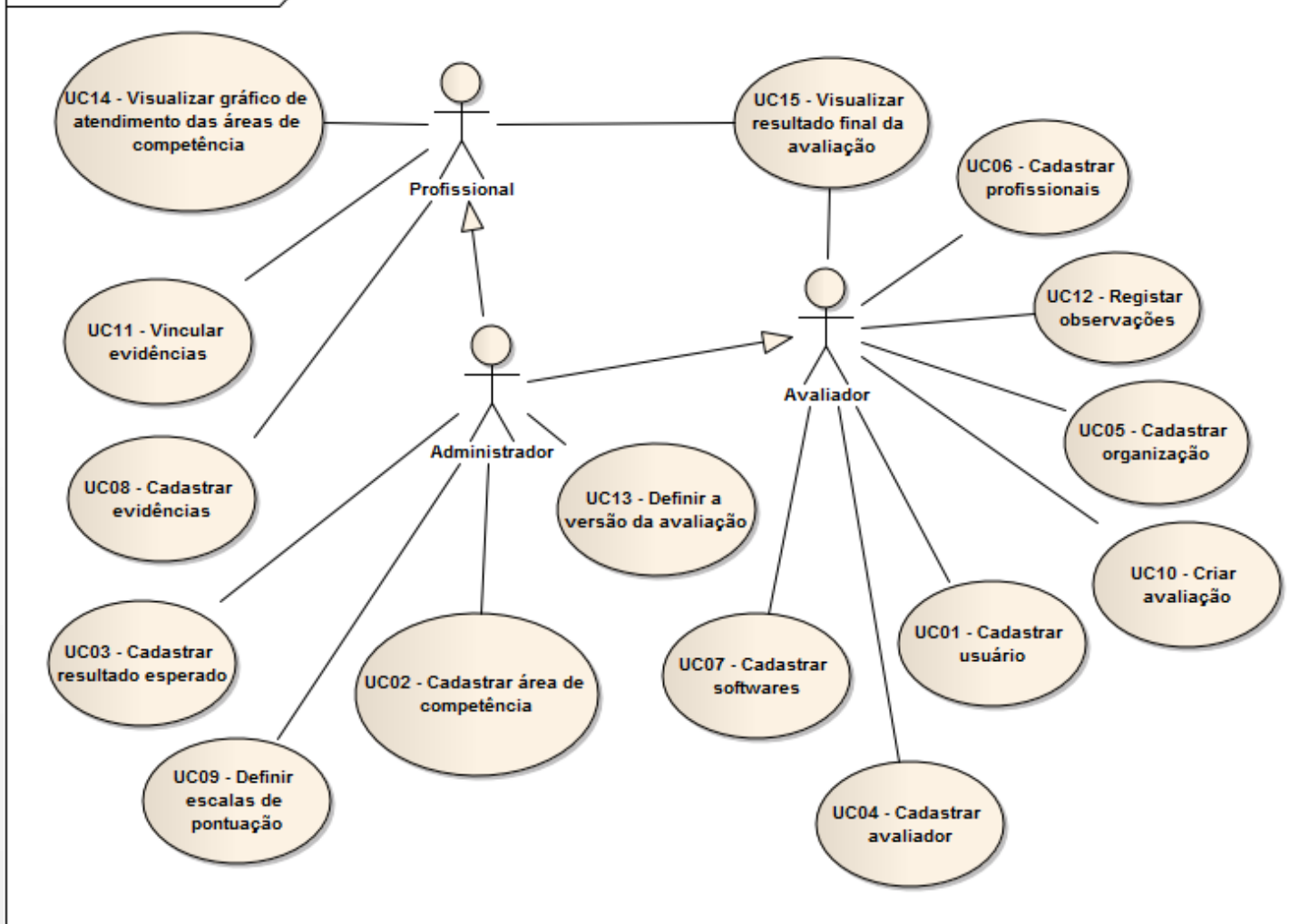
# Especificação

Foram utilizados os seguintes diagramas:

- Casos de uso;
- Classes;
- Atividades.
- Também foi especificado o Modelo Entidade Relacionamento (MER).

# Especificação

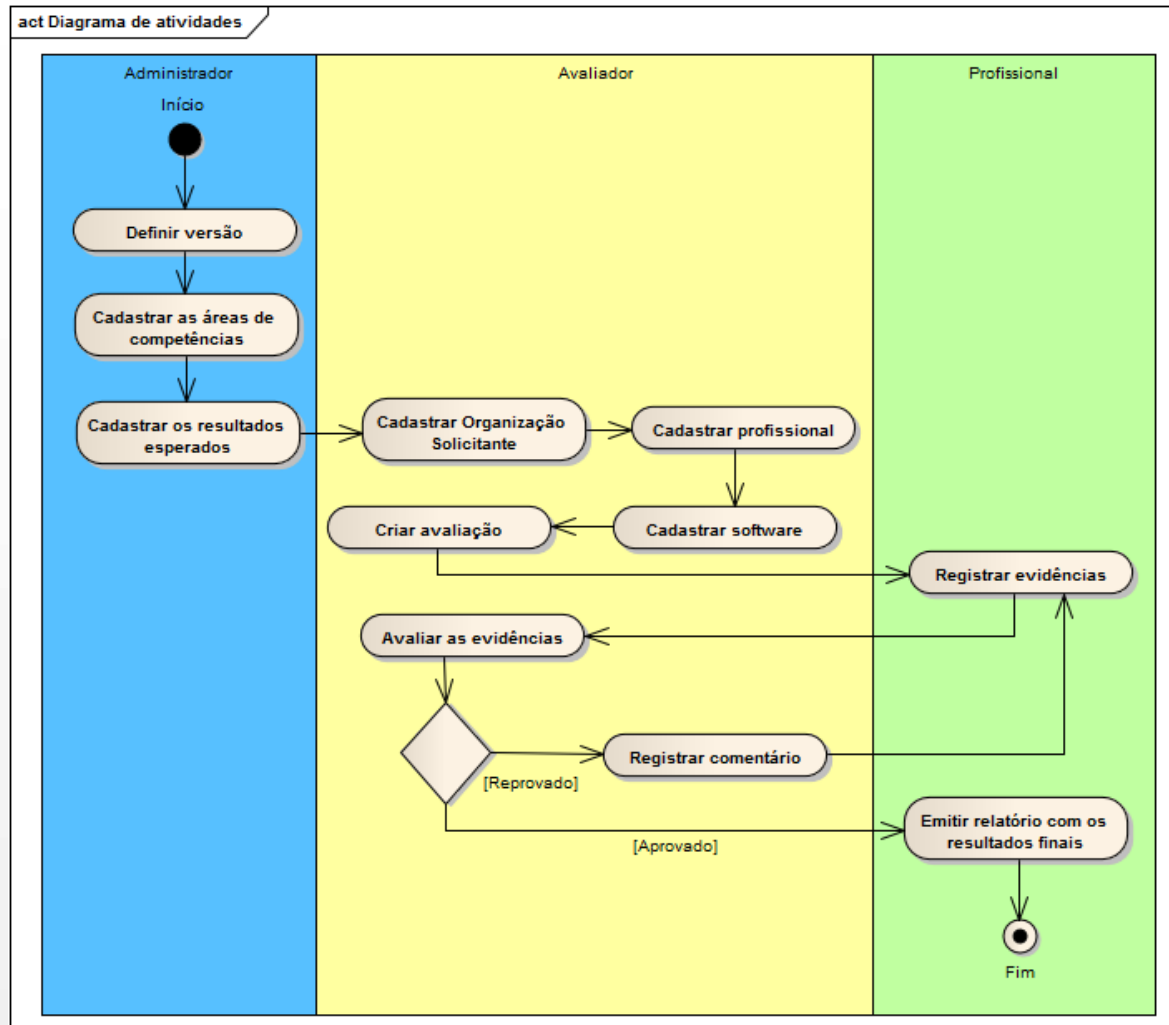
uc Diagrama de casos de uso



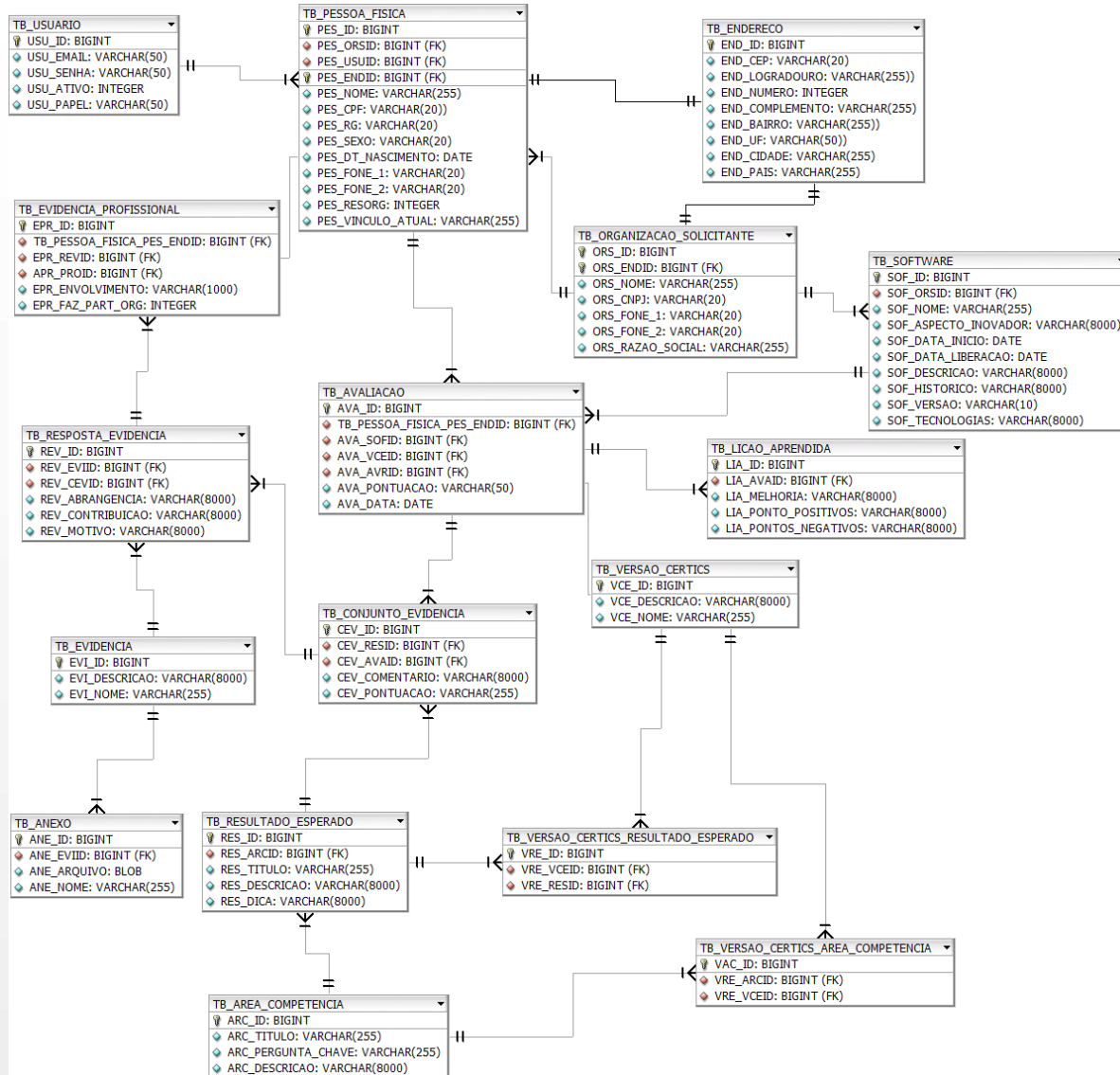
# Especificação

- Foram criados dois diagramas de classes.
- O primeiro tem por objetivo identificar as entidades são responsáveis pela lógica de negócio.
- O Segundo é referente à funcionalidade do vínculo de evidências ao software.

# Especificação



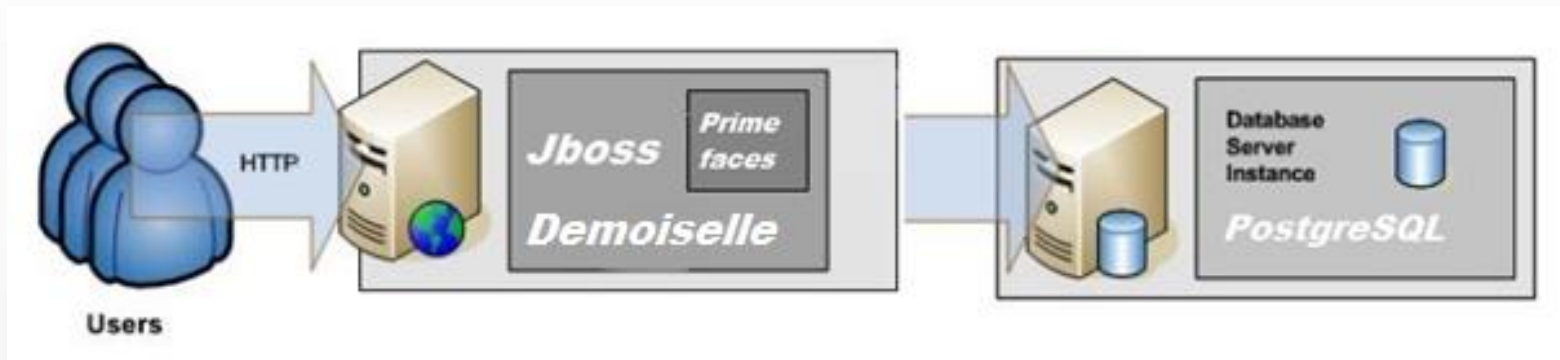
# Especificação



# Implementação

- Foi escolhida a linguagem JAVA na versão 7 para o desenvolvimento da ferramenta.
- Foram utilizadas as seguintes tecnologias:
  - *Framework Demoiselle*;
  - JBoss AS 7.1.1.Final;
  - PrimeFaces;
  - PostgreSQL;

# Implementação



# Implementação

- Para a construção da arquitetura da ferramenta foi seguido o padrão MVC.
- No *framework Demoiselle* este padrão é definido através da utilização dos seguintes estereótipos:
  - *@ViewController*;
  - *@BusinessController*;
  - *@PersistenceController*.



# Implementação

```
1 <p:layoutUnit id="treeRegistroEvidencias" position="west"
2   resizable="true" size="20%">
3     <p:tree id="tree" value="#{conjuntoEvidenciasEditMB.root}" var="node"
4       cache="false" selectionMode="single"
5       selection="#{conjuntoEvidenciasEditMB.selectedNode}"
6       style="width:100%;height:100%">
7
8       <p:ajax event="select" listener="#{conjuntoEvidenciasEditMB.onNodeSelect}"
9         update=":formRegistroEvidencias:panelPrincipalRegistroEvidencia,
10        :formRegistroEvidencias:panelPrincipalRegistroEvidenciaAvaliador,
11        :formRegistroEvidencias:panelPrincipalRegistroEvidenciaAvaliado,
12        :formRegistroEvidencias:panelPrincipalComentarioAvaliador,
13        :formRegistroEvidencias:dtEvidencias"/>
14
15     <p:treeNode id="treeNode">
16       <h:outputText id="lblNode" value="#{node.titulo}"/>
17     </p:treeNode>
18   </p:tree>
19 </p:layoutUnit>
```

# Implementação

```
1 public TreeNode getRoot(){
2     List<AreaCompetenciaEntity> lAreaCompetencia = this.areaCompetenciaDAO.
3     findByVersaoCerticsAndAvaliacaoID(this.getId(),
4     this.getBean().getVersaoCertics().getId());
5     this.root = new DefaultTreeNode("root", null);
6     for (AreaCompetenciaEntity areaCompetenciaEntity : lAreaCompetencia) {
7         TreeNode areaCompetencia = new DefaultTreeNode(new InformacoesArvore(
8         areaCompetenciaEntity.getId(), areaCompetenciaEntity.getTitulo(),
9         "areaCompetencia"), root);
10        for (ResultadoEsperadoEntity resultadoEsperadoEntity :
11        areaCompetenciaEntity.getResultadosEsperados()) {
12            TreeNode resultadoEsperado = new DefaultTreeNode(new InformacoesArvore(
13            resultadoEsperadoEntity.getId(), resultadoEsperadoEntity.getTitulo(),
14            "resultadoEsperado"), areaCompetencia);
15        }
16    }
17    return this.root;
18 }

19 public void onNodeSelect(NodeSelectEvent event) {
20     if("resultadoEsperado".equals(((InformacoesArvore) event.getTreeNode().getData
21     ()).getTipo())){
22         this.getBean().setConjuntoEvidenciasAux(this.conjuntoEvidenciasDAO.
23         findByResultadoEsperadoID(((InformacoesArvore) event.getTreeNode().getData
24         ()).getId()));
25     }
26 }
```

# Implementação

```
1 <h:panelGrid id="panelPrincipalRegistroEvidenciaAvaliador" columns="2"
2 rendered="#{conjuntoEvidenciasEditMB.mostraConjuntoEvidenciaAvaliador()}">
3   <h:outputLabel value="#{messages['conjuntoEvidencias.label.pontuacao']}: "
4     styleClass="text-input" />
5   <p:selectOneMenu id="pontuacao" effect="fade"
6     value="#{conjuntoEvidenciasEditMB.bean.conjuntoEvidenciasAux.pontuacao}">
7     <f:selectItems value="#{conjuntoEvidenciasEditMB.pontuacao}" />
8   </p:selectOneMenu>
9
10  <h:outputLabel value="#{messages['conjuntoEvidencias.label.comentario']}: "
11    for="conjuntoEvidenciasComentario" styleClass="text-input" />
12  <p:inputTextarea id="conjuntoEvidenciasComentario"
13    value="#{conjuntoEvidenciasEditMB.bean.conjuntoEvidenciasAux.comentario}"
14    title="#{messages['conjuntoEvidencias.alt.comentario']}" rows="10" cols="80"
15    maxLength="8000" autoResize="false"/>
16  <p:message for="conjuntoEvidenciasComentario" />
17
18  <p:commandButton value="#{messages['button.save']}"
19    actionListener="#{conjuntoEvidenciasEditMB.insertConjuntoEvidencia()}" />
20</h:panelGrid>
```

# Implementação

```
1 public void insertConjuntoEvidencia() {
2     this.conjuntoEvidenciasBC.update(this.getBean().getConjuntoEvidenciasAux());
3     atualizaPontuacaoAvaliacao();
4 }
5
6 private void atualizaPontuacaoAvaliacao() {
7     int qtdRespostas = 0, completamenteAtendido = 0, largamenteAtendido = 0,
8     parcialmenteAtendido = 0, naoAtendido = 0, naoRespondida = 0;
9     for (ConjuntoEvidenciasEntity conjuntoEvidenciasEntity :
10     conjuntoEvidenciasDAO.findByAvaliacaoID(getId())) {
11         qtdRespostas++;
12         if(conjuntoEvidenciasEntity.getPontuacao() != null){
13             switch (conjuntoEvidenciasEntity.getPontuacao()) {
14                 case F:
15                     completamenteAtendido++;
16                     break;
17                 case L:
18                     largamenteAtendido++;
19                     break;
20                 case P: parcialmenteAtendido++;
21                     break;
22                 case N: naoAtendido++;
23                     break;
24             }
25         }else{
26             naoRespondida++;
27         }
28     }
29     if(naoRespondida == 0 && (parcialmenteAtendido > 0 || naoAtendido > 0)){
30         getBean().setPontuacao(EnumPontuacaoAvaliacao.REPROVADA);
31     }else if(naoRespondida == 0 && qtdRespostas == (completamenteAtendido+
32     largamenteAtendido)){
33         getBean().setPontuacao(EnumPontuacaoAvaliacao.APROVADA);
34     }else{
35         getBean().setPontuacao(EnumPontuacaoAvaliacao.PENDENTE);
36     }
37     update();
38 }
```

# Operacionalidade da Implementação

Salvar Excluir

**Resultado esperado**

Código: 1

Área de competência: Área de Competência Desenvolvimento Tecnológico (DES)

Título: DES.1. Competência sobre Arquitetura

Descrição: A Unidade Organizacional tem competência sobre os elementos relevantes da arquitetura do software e sua implementação.

Dica:

- Definição do projeto de arquitetura do software;
- Projeto de arquitetura do software, documentado pelos profissionais da Unidade Organizacional;
- Capacitação dos profissionais da Unidade Organizacional nos resultados de um projeto de Pesquisa e Desenvolvimento Tecnológico (P&D) incorporado ao software, que foi executado em parceria com outras Organizações ou pela própria Organização;
- Aquisição de um componente relacionado à tecnologia relevante do software e incorporado na solução arquitetural;
- Decisão tomada pela Unidade Organizacional para a atualização da arquitetura adquirida;
- Capacitação dos profissionais da Unidade Organizacional que atuaram na arquitetura do software relacionada aos componentes tecnológicos relevantes, adquiridos ou desenvolvidos;

Versão da CERTICS:

Versão 1.0 →

→ Versão 1.1

↵

↶

# Operacionalidade da Implementação

## Evidência

Código: 1

Nome: Projeto de arquitetura do software

Descrição:

Definir de forma sistemática os requisitos do sistema envolvendo todos os stakeholders (interessados) no projeto de forma a ter o maior número possível de requisitos definidos de forma clara e objetiva;  
Os requisitos expressam as características e restrições do produto de software do ponto de vista de satisfação das necessidades do usuário, e, em geral independem da tecnologia empregada na construção da solução sendo a parte mais crítica e propensa a erros no desenvolvimento de software.  
Requisitos são objetivos ou restrições estabelecidas por clientes

## Anexo

### Novo

Código	Nome	Arquivo	Ação
1	Figura 1 - Avaliação para o modelo CMMI.png	+ Choose	Excluir
3	Evidencia.png	+ Choose	Excluir
4	arquitetura_sistema.gif	+ Choose	Excluir

# Operacionalidade da Implementação

Código: 2

Avaliador: Vinícius Fereda de Lima

Organização solicitante: TDV Systems

Nome: Scribim

Pontuação: Pendente

▼ Área de Competência Desen

DES.1. Competência sobre

DES.3. Fases e Disciplinas

DES.6. Competência para :

DES.2. Competência sobre

DES.4. Papéis e Pessoas Ic

DES.5. Dados Técnicos Rel

▶ Área de Competência Gestã

▶ Área de Competência Gestã

▶ Área de Competência Melho

**Sim**

**Não**

**Não sei, preciso de ajuda**

Pontuação: Largamente atendido

Comentário: Foram registrados dois artefatos muito bons. Porém poderia ter um pouco mais de detalhamento.

## Evidência

Nome	Abrangência	Contribuição	Motivo	Profissionais	Ações
<a href="#">Lista dos profissionais contratados no regime CLT</a>	Indicador de que a propriedade intelectual do software desenvolvido por seus profissionais, no âmbito do contrato de trabalho, pertence à Organização	Lista dos profissionais contratados no regime CLT da Unidade Organizacional que atuam na arquitetura do software relacionada aos componentes tecnológicos relevantes, adquiridos ou desenvolvidos.		<b>Novo</b> <b>Listar</b>	<b>Editar</b> <b>Excluir</b>
<a href="#">Projeto de arquitetura do software</a>	Ela facilita o entendimento por parte do interessado, uma vez que vai filtrar e formatar a informação.	Enfatizar a importância da arquitetura para o sucesso de um projeto de software. A visão fornecida pelos casos de uso do sistema, pode interessar ao cliente/usuário.		<b>Novo</b> <b>Listar</b>	<b>Editar</b> <b>Excluir</b>

# Operacionalidade da Implementação

## Gráfico de atendimento das áreas de competência

Área de competência: Área de Competência Desenvolvimento Tecnológico (DES)

### Resultado esperado

DES.1. Competência sobre Arquitetura

DES.2. Competência sobre Requisitos

DES.3. Fases e Disciplinas Compatíveis com o Software

DES.4. Papéis e Pessoas Identificados

DES.5. Dados Técnicos Relevantes Documentados

DES.6. Competência para Suporte e Evolução do Software

### Atendido?

Sim

Sim

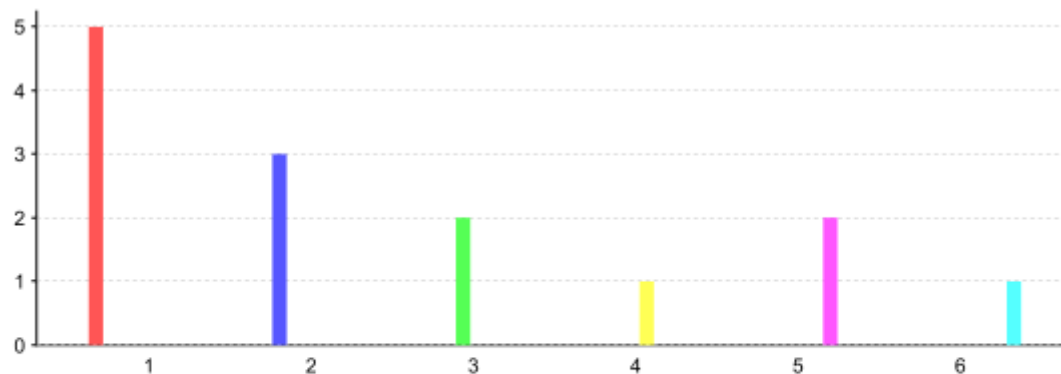
Sim

Sim

Sim

Sim

### Quantidade de evidências



■ DES.1. Competência sobre Arquitetura ■ DES.2. Competência sobre Requisitos  
■ DES.3. Fases e Disciplinas Compatíveis com o Software ■ DES.4. Papéis e Pessoas Identificados  
■ DES.5. Dados Técnicos Relevantes Documentados ■ DES.6. Competência para Suporte e Evolução do Software



# Resultados e Discussões

- Tornou-se possível para o avaliador criar combinações de versões diferentes sem que seja necessário seguir as versões padrões disponibilizadas pela metodologia CERTICS.
- Tornou a avaliação mais flexível.
- São avaliadas as regras definidas pela metodologia CERTICS.

# Resultados e Discussões

<b>FUNCIONALIDADES</b>	<b>TCC</b>	<b>Albuquerque Júnior</b>	<b>Borges</b>	<b>CERTICSys</b>
Metodologia CERTICS	X			X
Ambiente web	X		X	X
Evidências com anexo	X	X	X	X
Relatórios gerenciais	X	X	X	
Portal aberto para pessoas físicas	X	X	X	
Tutor automatizado que auxilia na identificação das evidências				X

# Conclusões e Sugestões

- O principal aspecto a ser apresentado é que a ferramenta utilizará a metodologia CERTICS para a realização da avaliação do software
- Maior facilidade ao avaliador para que as evidências sejam devidamente cadastradas e avaliadas.

# Conclusões e Sugestões

- Evita-se que o avaliador não tenha que fazer uso de outros softwares como editores de texto e planilhas de cálculo.
- Tende a diminuir o tempo gasto pelo avaliador e a organização avaliada.
- Pode-se utilizar a ferramenta para dinamizar o ensino de qualidade de software.

# Conclusões e Sugestões

- Construir um recurso que gere as dicas dos resultados esperados com base nas evidências já cadastradas na ferramenta.
- Criar um fórum que possibilite o avaliador e a organização solicitante se comunicar dentro da ferramenta.
- Disponibilizar um mecanismo que realize uma avaliação prévia automaticamente das evidências vinculadas.

**Obrigado!**